NCTU CN2018 Lab. 1 – Packet Manipulation via Scapy

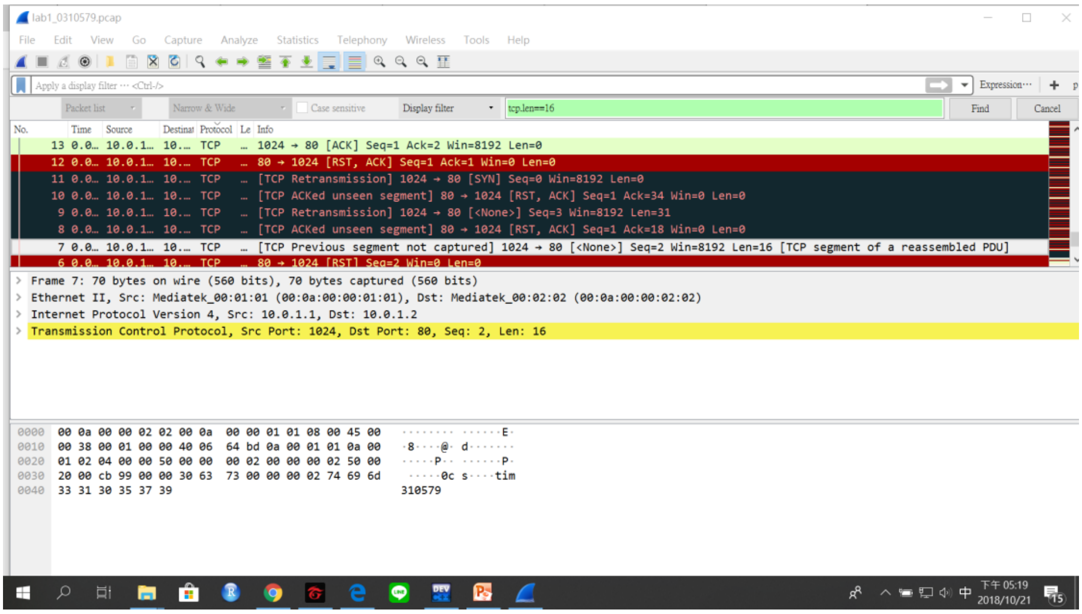
Student name: 陳昱銘 Student ID: 0616027 Department: CS

Part A. Questions

1. What is your command to filter the packet with customized header on Wireshark?

過濾 tcp.len 是等於 16，因為其他都是零或 31(secret)

2. Show the screenshot of filtering the packet with customized header.



Display filter: tcp.len==16

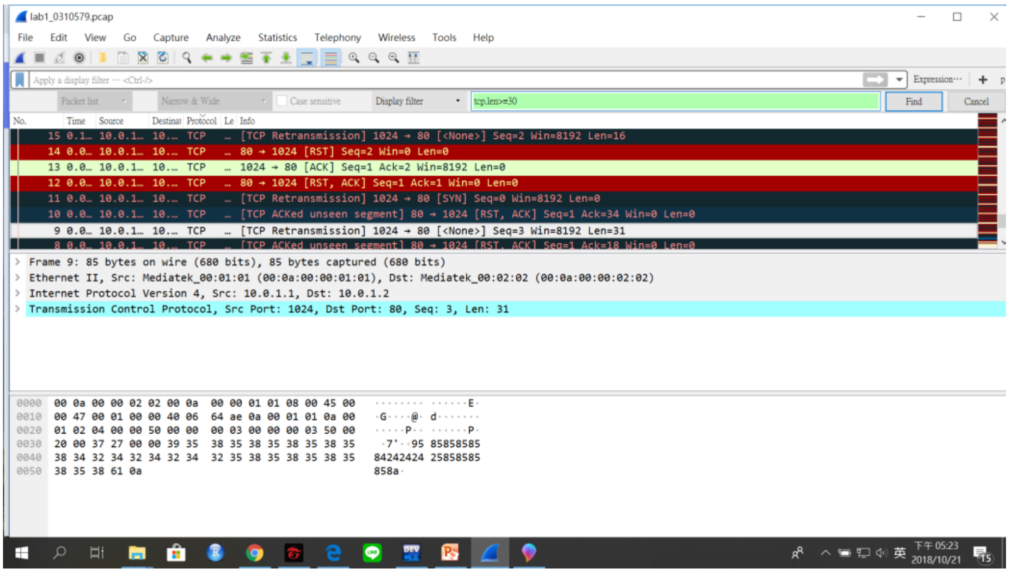
```
No.  Time  Source    Destinat Protocol Le Info
 13 0.0… 10.0.1… 10.… TCP … 1024 → 80 [ACK] Seq=1 Ack=2 Win=8192 Len=0
 12 0.0… 10.0.1… 10.… TCP … 80 → 1024 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
 11 0.0… 10.0.1… 10.… TCP … [TCP Retransmission] 1024 → 80 [SYN] Seq=0 Win=8192 Len=0
 10 0.0… 10.0.1… 10.… TCP … [TCP ACKed unseen segment] 80 → 1024 [RST, ACK] Seq=1 Ack=34 Win=0 Len=0
  9 0.0… 10.0.1… 10.… TCP … [TCP Retransmission] 1024 → 80 [<None>] Seq=3 Win=8192 Len=31
  8 0.0… 10.0.1… 10.… TCP … [TCP ACKed unseen segment] 80 → 1024 [RST, ACK] Seq=1 Ack=18 Win=0 Len=0
  7 0.0… 10.0.1… 10.… TCP … [TCP Previous segment not captured] 1024 → 80 [<None>] Seq=2 Win=8192 Len=16 [TCP segment of a reassembled PDU]
  6 0.0… 10.0.1… 10.… TCP … 80 → 1024 [RST] Seq=2 Win=0 Len=0

> Frame 7: 70 bytes on wire (560 bits), 70 bytes captured (560 bits)
> Ethernet II, Src: Mediatek_00:01:01 (00:0a:00:00:01:01), Dst: Mediatek_00:02:02 (00:0a:00:00:02:02)
> Internet Protocol Version 4, Src: 10.0.1.1, Dst: 10.0.1.2
> Transmission Control Protocol, Src Port: 1024, Dst Port: 80, Seq: 2, Len: 16
```

```
0000  00 0a 00 00 02 02 00 0a  00 00 01 01 08 00 45 00   ········ ······E·
0010  00 38 00 01 00 00 40 06  64 bd 0a 00 01 01 0a 00   ·8····@· d·······
0020  01 02 04 00 00 50 00 00  00 02 00 00 00 02 50 00   ·····P·· ······P·
0030  20 00 cb 99 00 00 30 63  73 00 00 00 02 74 69 6d    ·····0c s····tim
0040  33 31 30 35 37 39                                  310579
```

3. What is your command to filter the packet with “secret” payload on Wireshark?

過濾 tcp.len 是大於等於 30，其他的都小於 30

4. Show the screenshot of filtering the packet with “secret” payload.

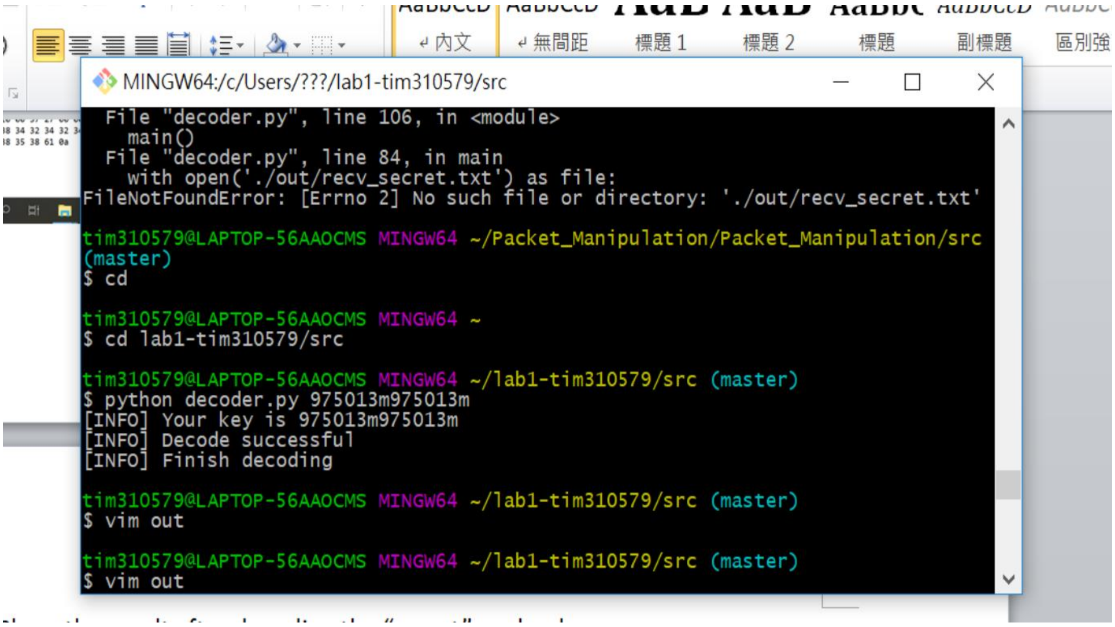


Display filter: tcp.len>=30

```
No.  Time  Source    Destinat Protocol Le Info
 15 0.1… 10.0.1… 10.… TCP … [TCP Retransmission] 1024 → 80 [<None>] Seq=2 Win=8192 Len=16
 14 0.0… 10.0.1… 10.… TCP … 80 → 1024 [RST] Seq=2 Win=0 Len=0
 13 0.0… 10.0.1… 10.… TCP … 1024 → 80 [ACK] Seq=1 Ack=2 Win=8192 Len=0
 12 0.0… 10.0.1… 10.… TCP … 80 → 1024 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
 11 0.0… 10.0.1… 10.… TCP … [TCP Retransmission] 1024 → 80 [SYN] Seq=0 Win=8192 Len=0
 10 0.0… 10.0.1… 10.… TCP … [TCP ACKed unseen segment] 80 → 1024 [RST, ACK] Seq=1 Ack=34 Win=0 Len=0
  9 0.0… 10.0.1… 10.… TCP … [TCP Retransmission] 1024 → 80 [<None>] Seq=3 Win=8192 Len=31
  8 0.0… 10.0.1… 10.… TCP … [TCP ACKed unseen segment] 80 → 1024 [RST, ACK] Seq=1 Ack=18 Win=0 Len=0

> Frame 9: 85 bytes on wire (680 bits), 85 bytes captured (680 bits)
> Ethernet II, Src: Mediatek_00:01:01 (00:0a:00:00:01:01), Dst: Mediatek_00:02:02 (00:0a:00:00:02:02)
> Internet Protocol Version 4, Src: 10.0.1.1, Dst: 10.0.1.2
> Transmission Control Protocol, Src Port: 1024, Dst Port: 80, Seq: 3, Len: 31
```

```
0000  00 0a 00 00 02 02 00 0a  00 00 01 01 08 00 45 00   ········ ······E·
0010  00 47 00 01 00 00 40 06  64 ae 0a 00 01 01 0a 00   ·G····@· d·······
0020  01 02 04 00 00 50 00 00  00 03 00 00 00 03 50 00   ·····P·· ······P·
0030  20 00 37 27 00 00 39 35  38 35 38 35 38 35 38 35    ·7'··95 85858585
0040  38 34 32 34 32 34 32 34  32 35 38 35 38 35 38 35   84242424 25858585
0050  38 35 38 61 0a                                     858a·
```

5. Show the result after decoding the “secret” payload.



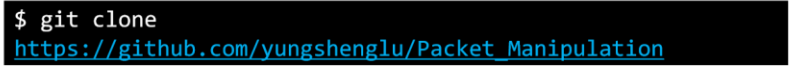
```
  File "decoder.py", line 106, in <module>
    main()
  File "decoder.py", line 84, in main
    with open('./out/recv_secret.txt') as file:
FileNotFoundError: [Errno 2] No such file or directory: './out/recv_secret.txt'

tim310579@LAPTOP-56AAOCMS MINGW64 ~/Packet_Manipulation/Packet_Manipulation/src
(master)
$ cd

tim310579@LAPTOP-56AAOCMS MINGW64 ~
$ cd lab1-tim310579/src

tim310579@LAPTOP-56AAOCMS MINGW64 ~/lab1-tim310579/src (master)
$ python decoder.py 975013m975013m
[INFO] Your key is 975013m975013m
[INFO] Decode successful
[INFO] Finish decoding

tim310579@LAPTOP-56AAOCMS MINGW64 ~/lab1-tim310579/src (master)
$ vim out

tim310579@LAPTOP-56AAOCMS MINGW64 ~/lab1-tim310579/src (master)
$ vim out
```

Part B. Description

Task 1 – Environment setup

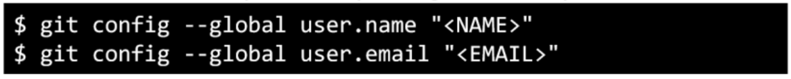
先把檔案都先下載下來，再存到自己的帳戶儲存處中

- Download required files from GitHub



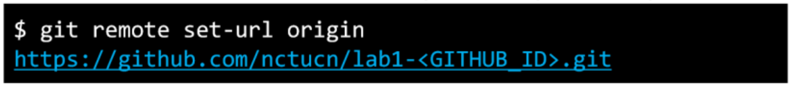
```
$ git clone
https://github.com/yungshenglu/Packet_Manipulation
```

- Get and set repository or global options



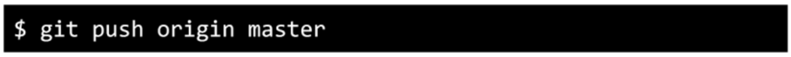
```
$ git config --global user.name "<NAME>"
$ git config --global user.email "<EMAIL>"
```

- Set a new remote URL to your repository



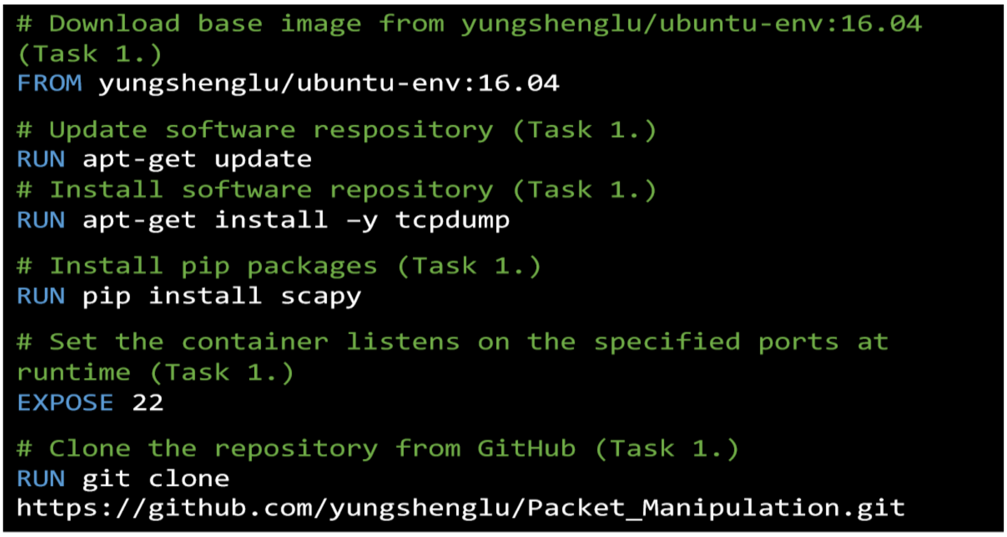
```
$ git remote set-url origin
https://github.com/nctucn/lab1-<GITHUB_ID>.git
```

- Push your repository to remote



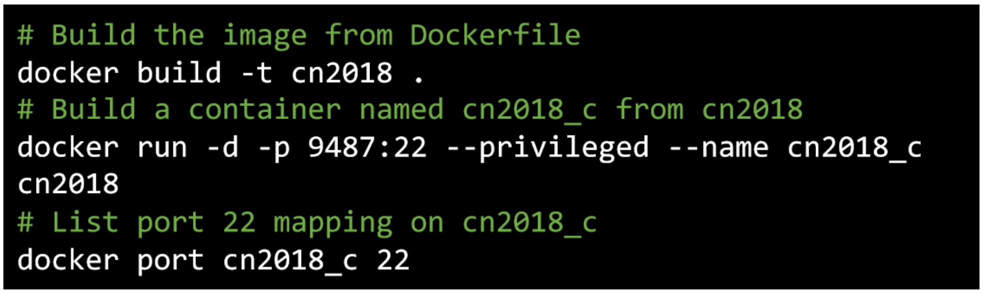
```
$ git push origin master
```

從 yungshenglu/ubuntu-env:16.04 下載圖片，更新再安裝軟體，捨定一下 container，再從 github 複製 repository



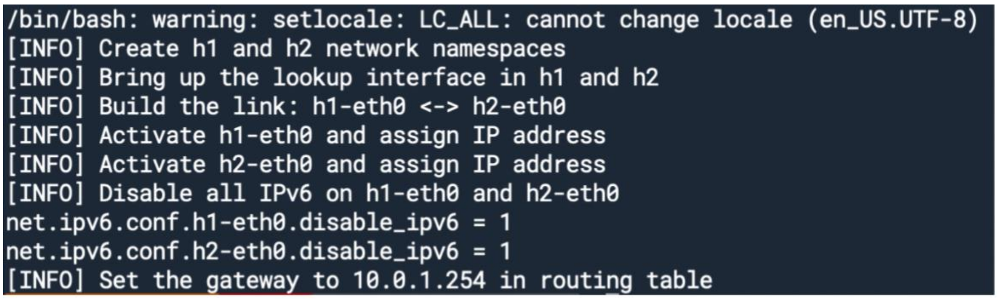
```
# Download base image from yungshenglu/ubuntu-env:16.04
(Task 1.)
FROM yungshenglu/ubuntu-env:16.04

# Update software respository (Task 1.)
RUN apt-get update
# Install software repository (Task 1.)
RUN apt-get install -y tcpdump

# Install pip packages (Task 1.)
RUN pip install scapy

# Set the container listens on the specified ports at
runtime (Task 1.)
EXPOSE 22

# Clone the repository from GitHub (Task 1.)
RUN git clone
https://github.com/yungshenglu/Packet_Manipulation.git
```

打開 CMD 改變路徑再建造環境



```
# Build the image from Dockerfile
docker build -t cn2018 .
# Build a container named cn2018_c from cn2018
docker run -d -p 9487:22 --privileged --name cn2018_c
cn2018
# List port 22 mapping on cn2018_c
docker port cn2018_c 22
```

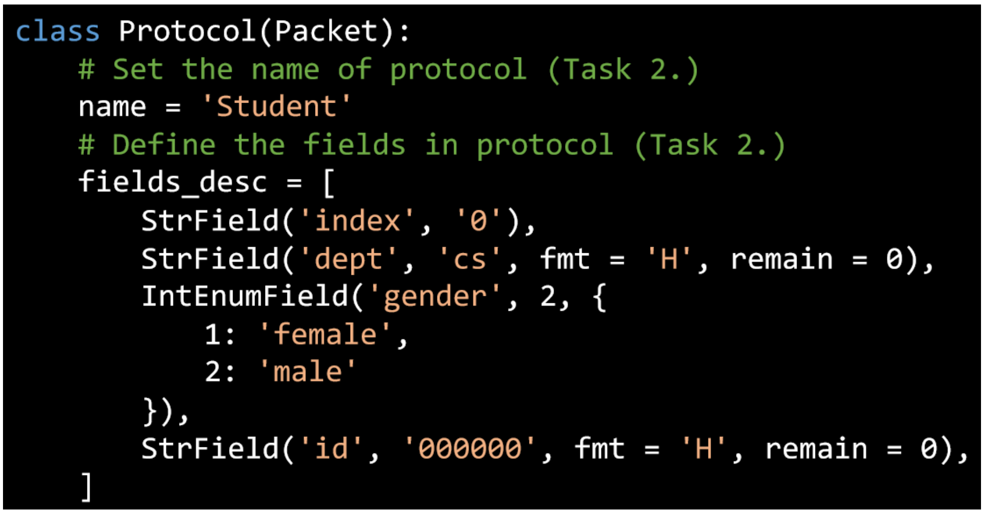
之後再打開 pietty 連到 docker，再為 h2 創造命名空間



```
/bin/bash: warning: setlocale: LC_ALL: cannot change locale (en_US.UTF-8)
[INFO] Create h1 and h2 network namespaces
[INFO] Bring up the lookup interface in h1 and h2
[INFO] Build the link: h1-eth0 <-> h2-eth0
[INFO] Activate h1-eth0 and assign IP address
[INFO] Activate h2-eth0 and assign IP address
[INFO] Disable all IPv6 on h1-eth0 and h2-eth0
net.ipv6.conf.h1-eth0.disable_ipv6 = 1
net.ipv6.conf.h2-eth0.disable_ipv6 = 1
[INFO] Set the gateway to 10.0.1.254 in routing table
```

Task 2. Define protocol via Scapy

自己定義 protocol



```
class Protocol(Packet):
    # Set the name of protocol (Task 2.)
    name = 'Student'
    # Define the fields in protocol (Task 2.)
    fields_desc = [
        StrField('index', '0'),
        StrField('dept', 'cs', fmt = 'H', remain = 0),
        IntEnumField('gender', 2, {
            1: 'female',
            2: 'male'
        }),
        StrField('id', '000000', fmt = 'H', remain = 0),
    ]
```

Task 3. Send packets

把 sender 的 code 寫好

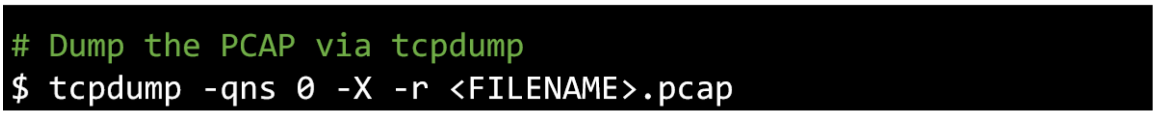
Task 4. Sniff packets

把 receiver 的 code 寫好

Task 5. Run sender and receiver

打開 tmux，並把視窗分為兩邊，一邊寄另一邊收，(先跑 receiver 再跑 sender)

再用 tcpdump show 出 Pcap file



```
# Dump the PCAP via tcpdump
$ tcpdump -qns 0 -X -r <FILENAME>.pcap
```

收完 packet 之後就會拿到 Pcap file 和 recv_secret.txt

Task 6. Push your files to remote

把圖片上傳到 docker hub

```
# Create a new image from a container’s changes
$ docker commit cn2018_c <DOCKER_HUB_ID>/cn2018_lab1
# Login to your Docker registry
$ docker login
# Push an image to a registry
$ docker push <DOCKER_HUB_ID>/cn2018_lab1
```

檔案上傳到 github

```
# Get and set repository or global options
$ git config --global user.name "<NAME>"
$ git config --global user.email "<EMAIL>"
# Add your files into staging area
$ git add .
# Commit your files
$ git commit -m "Commit lab1 in class”
# Set the remote URL to your remote repository
$ git remote set-url origin
https://github.com/nctucn/lab1-<YOUR_ID>.git
# Push your files to remote repository
$ git push origin master
```

Task 7. Load PCAP via Wireshark

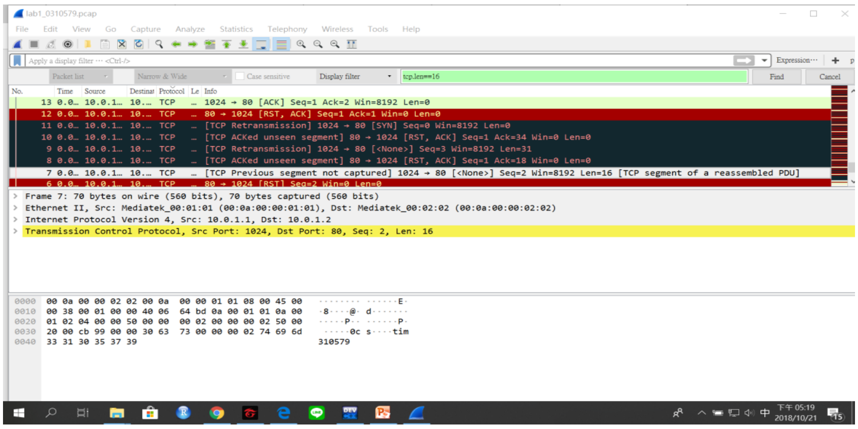
用 wireshark 打開 Pcap file

Task 8. Filter the target packet

過濾這些封包，裡面有 protocol，還有 secret key 之類的，之後找出 14 個 secret

Key

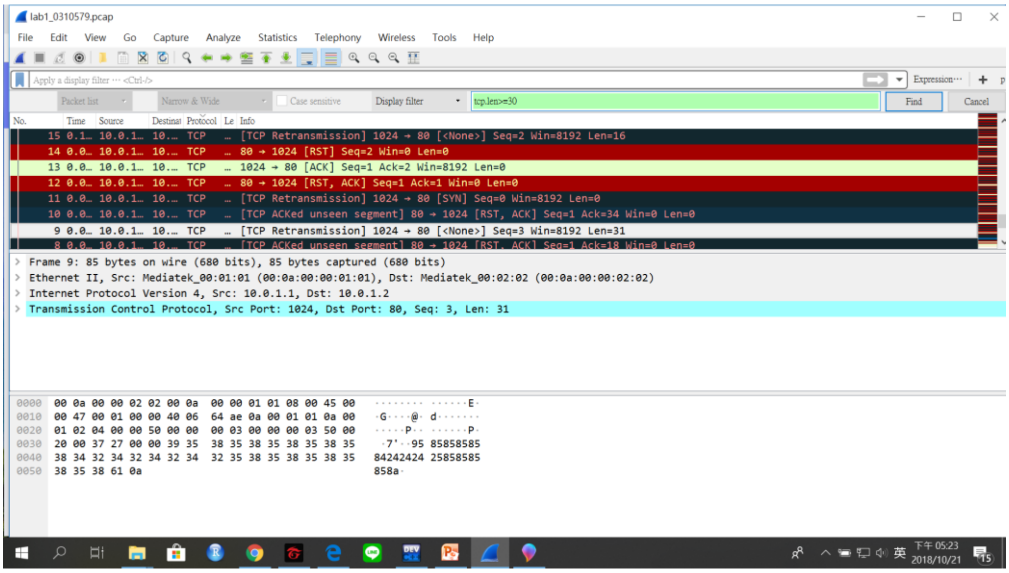
Filter the packets of our defined protocol: 過濾 tcp.len 是等於 16，因為其他都是零或 31(secret)



Display filter: tcp.len==16

```
No.   Time   Source    Destinat Protocol Le Info
   13 0.0…  10.0.1…  10.…  TCP  …  1024 → 80 [ACK] Seq=1 Ack=2 Win=8192 Len=0
   12 0.0…  10.0.1…  10.…  TCP  …  80 → 1024 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
   11 0.0…  10.0.1…  10.…  TCP  …  [TCP Retransmission] 1024 → 80 [SYN] Seq=0 Win=8192 Len=0
   10 0.0…  10.0.1…  10.…  TCP  …  [TCP ACKed unseen segment] 80 → 1024 [RST, ACK] Seq=1 Ack=34 Win=0 Len=0
    9 0.0…  10.0.1…  10.…  TCP  …  [TCP Retransmission] 1024 → 80 [<None>] Seq=3 Win=8192 Len=31
    8 0.0…  10.0.1…  10.…  TCP  …  [TCP ACKed unseen segment] 80 → 1024 [RST, ACK] Seq=1 Ack=18 Win=0 Len=0
    7 0.0…  10.0.1…  10.…  TCP  …  [TCP Previous segment not captured] 1024 → 80 [<None>] Seq=2 Win=8192 Len=16 [TCP segment of a reassembled PDU]
    6 0.0…  10.0.1…  10.…  TCP  …  80 → 1024 [RST] Seq=2 Win=0 Len=0

> Frame 7: 70 bytes on wire (560 bits), 70 bytes captured (560 bits)
> Ethernet II, Src: Mediatek_00:01:01 (00:0a:00:00:01:01), Dst: Mediatek_00:02:02 (00:0a:00:00:02:02)
> Internet Protocol Version 4, Src: 10.0.1.1, Dst: 10.0.1.2
> Transmission Control Protocol, Src Port: 1024, Dst Port: 80, Seq: 2, Len: 16

0000  00 0a 00 00 02 02 00 0a  00 00 01 01 08 00 45 00   ········ ······E·
0010  00 38 00 01 00 00 40 06  64 bd 0a 00 01 01 0a 00   ·8····@· d·······
0020  01 02 04 00 00 50 00 00  00 02 00 00 00 02 50 00   ·····P·· ······P·
0030  20 00 cb 99 00 00 30 63  73 00 00 00 02 74 69 6d    ·····0c s····tim
0040  33 31 30 35 37 39                                  310579
```

Filter the packets with the “secret” bits: 過濾 tcp.len 是大於等於 30，其他的都小於 30



```
No.  Time   Source    Destinat Protocol Le Info
  15 0.1…  10.0.1…   10.…     TCP      …  [TCP Retransmission] 1024 → 80 [<None>] Seq=2 Win=8192 Len=16
  14 0.0…  10.0.1…   10.…     TCP      …  80 → 1024 [RST] Seq=2 Win=0 Len=0
  13 0.0…  10.0.1…   10.…     TCP      …  1024 → 80 [ACK] Seq=1 Ack=2 Win=8192 Len=0
  12 0.0…  10.0.1…   10.…     TCP      …  80 → 1024 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
  11 0.0…  10.0.1…   10.…     TCP      …  [TCP Retransmission] 1024 → 80 [SYN] Seq=0 Win=8192 Len=0
  10 0.0…  10.0.1…   10.…     TCP      …  [TCP ACKed unseen segment] 80 → 1024 [RST, ACK] Seq=1 Ack=34 Win=0 Len=0
   9 0.0…  10.0.1…   10.…     TCP      …  [TCP Retransmission] 1024 → 80 [<None>] Seq=3 Win=8192 Len=31
   8 0.0…  10.0.1…   10.…     TCP      …  [TCP ACKed unseen segment] 80 → 1024 [RST, ACK] Seq=1 Ack=18 Win=0 Len=0

> Frame 9: 85 bytes on wire (680 bits), 85 bytes captured (680 bits)
> Ethernet II, Src: Mediatek_00:01:01 (00:0a:00:00:01:01), Dst: Mediatek_00:02:02 (00:0a:00:00:02:02)
> Internet Protocol Version 4, Src: 10.0.1.1, Dst: 10.0.1.2
> Transmission Control Protocol, Src Port: 1024, Dst Port: 80, Seq: 3, Len: 31

0000  00 0a 00 00 02 02 00 0a  00 00 01 01 08 00 45 00   ········ ······E·
0010  00 47 00 01 00 00 40 06  64 ae 0a 00 01 01 0a 00   ·G····@· d·······
0020  01 02 04 00 00 50 00 00  00 03 00 00 00 03 50 00   ·····P·· ······P·
0030  20 00 37 27 00 00 39 35  38 35 38 35 38 35 38 35    ·7'··95 85858585
0040  38 34 32 34 32 34 32 34  32 35 38 35 38 35 38 35   84242424 25858585
0050  38 35 38 61 0a                                      858a·
```

總共有 14 個 secret 的封包：我的 secret key 是 975013m975013m，14 個封包都看過一次才找完，後來發現好像是我的 github 帳號後七碼倒過來寫兩次而已。

Task 9. Decode the secret key

打開 git bash，更新過 decoder.py 後，進到 src 資料夾，並 decode 我的 secret key，並成功 decode，之後再把檔案都上傳到 github。

What you have learned in this lab?

這次的 lab，我認識了很多之前沒見過的工具，像是 github、docker、scapy 等等，學習了 C++以外的許多語法，像是基本的 cd 、vim 等，或是一些 python 等等的指令，另外也體會到網路的封包傳送過程。

What difficulty you have met in this lab?

光是 lab 就弄了快五個小時，因為對語法的不熟悉，有些指令也不知道要打在哪裡，很多種工具也不曉得要開哪一個，想問助教也要等很久，在一些厲害的同學 carry 之下，最後總算是做出了結果，本來以為回家後要做的事情

應該不會太難，結果發現 wireshark 的過濾條件不知道要打什麼，看了好久才發現剛好有 14 個 packet 的 len 是 16，才想到可以利用 len 來過濾，找出 secret key 後要執行 decoder，才發現自己的電腦的 git bash 很多指令都不能用，還要另外安裝一些東西，像是 python，就還要另外設置變量才能使用，還有一些 PIL 的問題等等，google 了好久才找到答案，我覺得這次的 lab 雖然讓我遇到了很多困難感到很挫折，但是解決問題後意外地讓人神清氣爽，我在這次 lab 也學到了很多解決問題的能力。